

# How To Get <sup>Exactly</sup> What You Want From Developer(s)

Steve Glinberg

Founder, 123 Apps

steve@123ColorApp.com

@123ColorAppDev

AppCamp

May 23, 2011

Artwork by Nate Theis

# Background



KidCalc Math Fun



123 Color



123 World



123 Glow



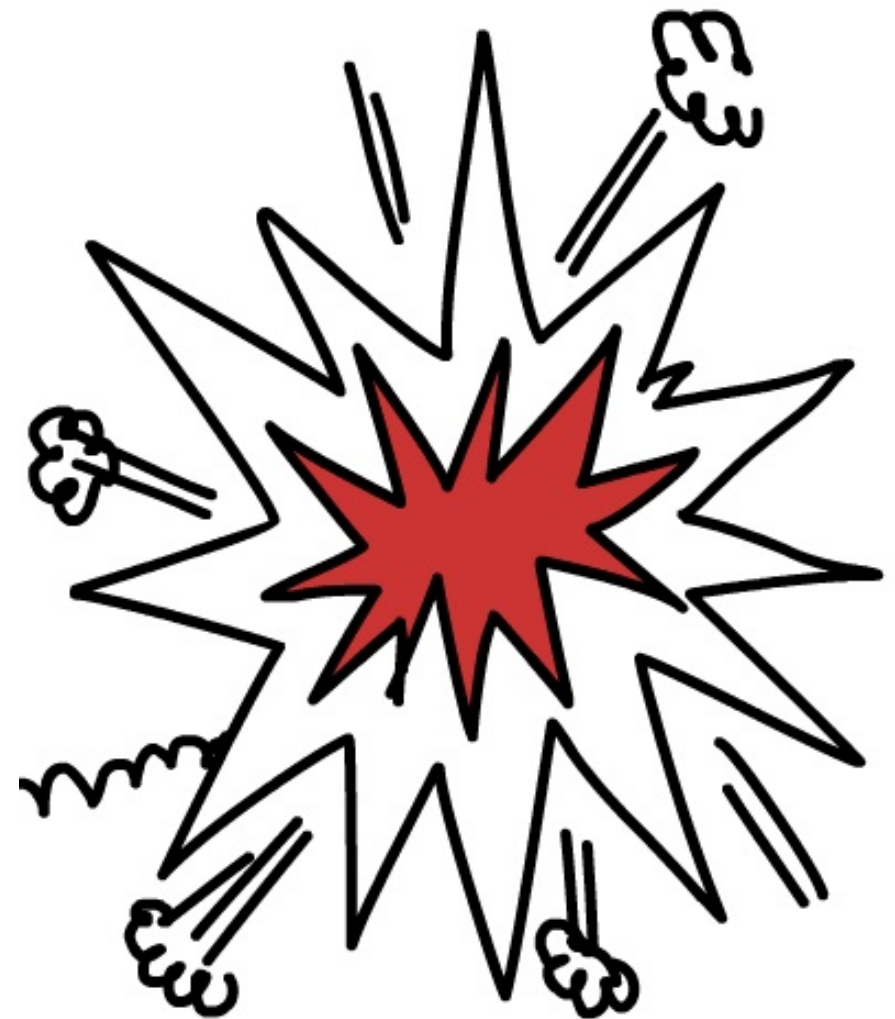
123 Sticker



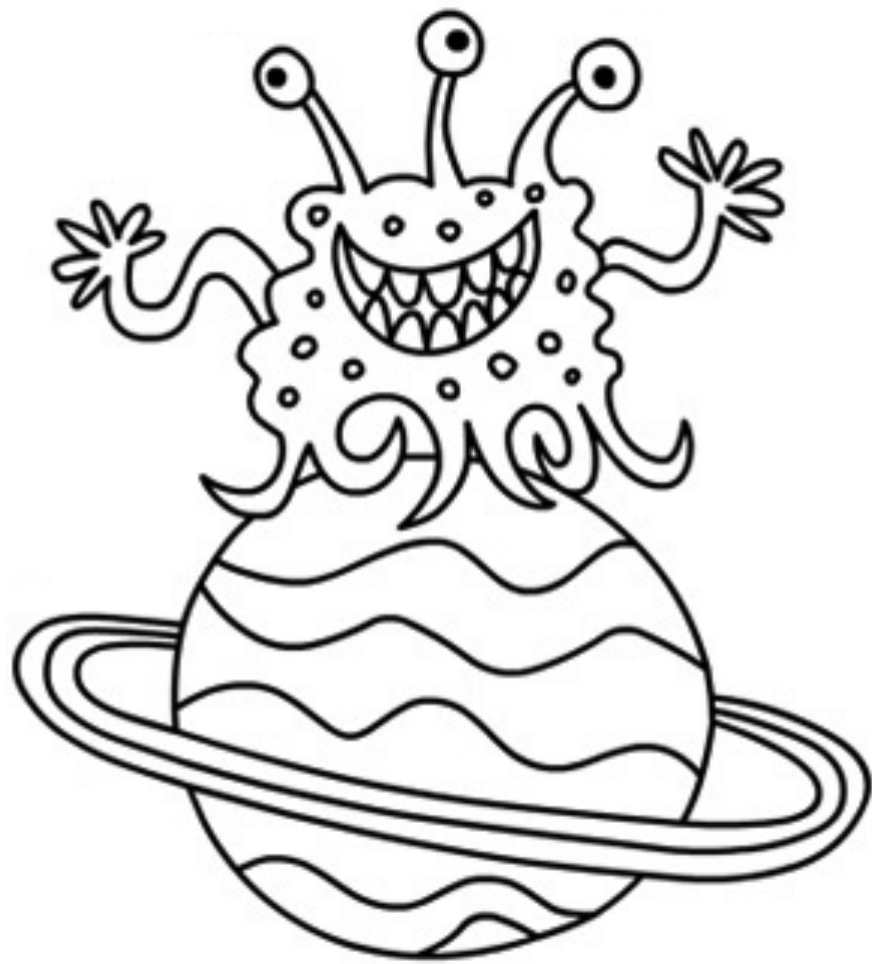
123 Color  
Premium Edition

# Lesson #1: Common Causes of Disasters

- Missing, Unclear, or Overlooked Requirements
- Invalid Assumptions
- Priorities Unstated or Unclear



# Lesson #2: Developers are Very Interesting Creatures



In Contrast to Many Humanoids,  
Developers:

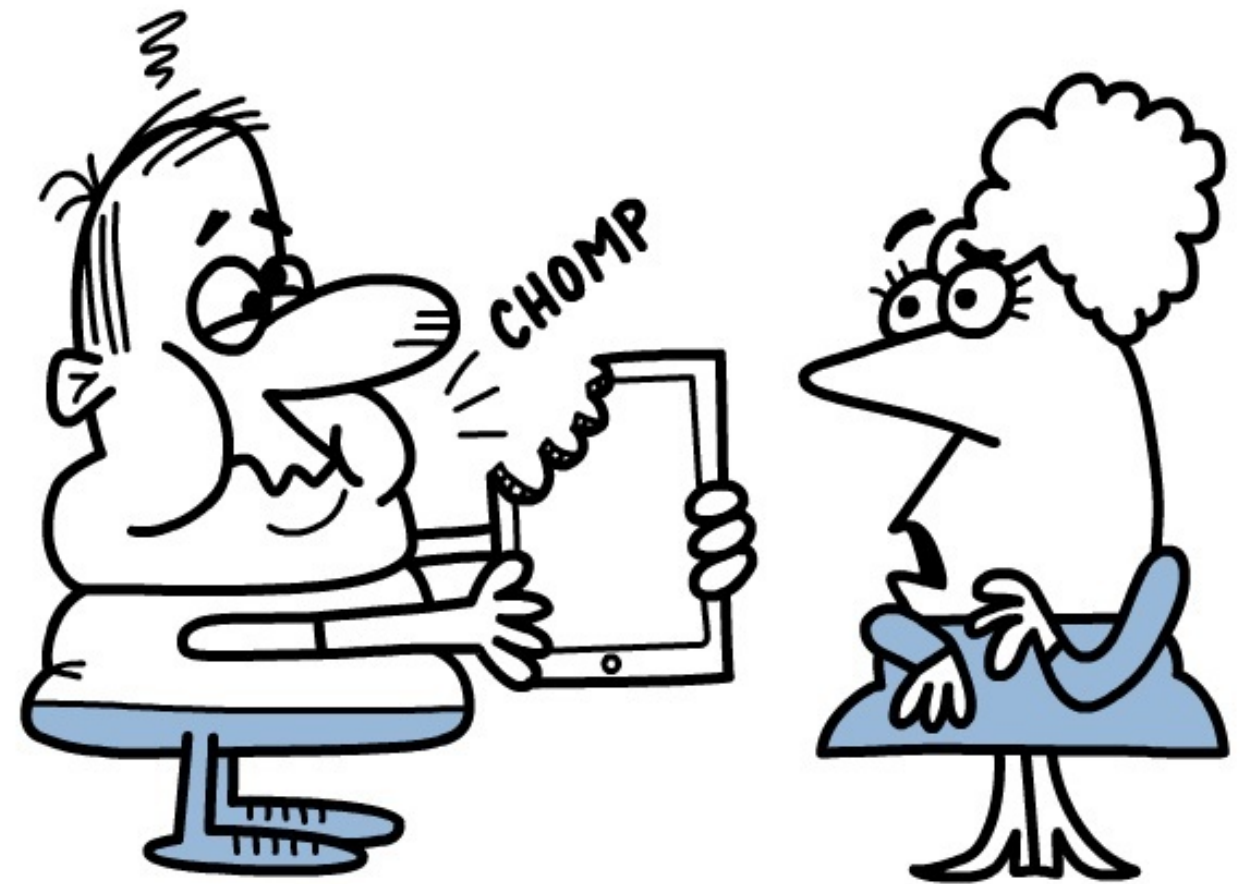
- Can Be Extremely Analytical, Literal.

Like Many Humanoids, Developers:

- Can Make Invalid Assumptions
- Are Influenced by Their Own Biases
- May Not Remember What You Tell Them.

# Lesson #3: Good Things Happen When You Feed Your Developers

- The Information They Need.
- In the Format They Need It.

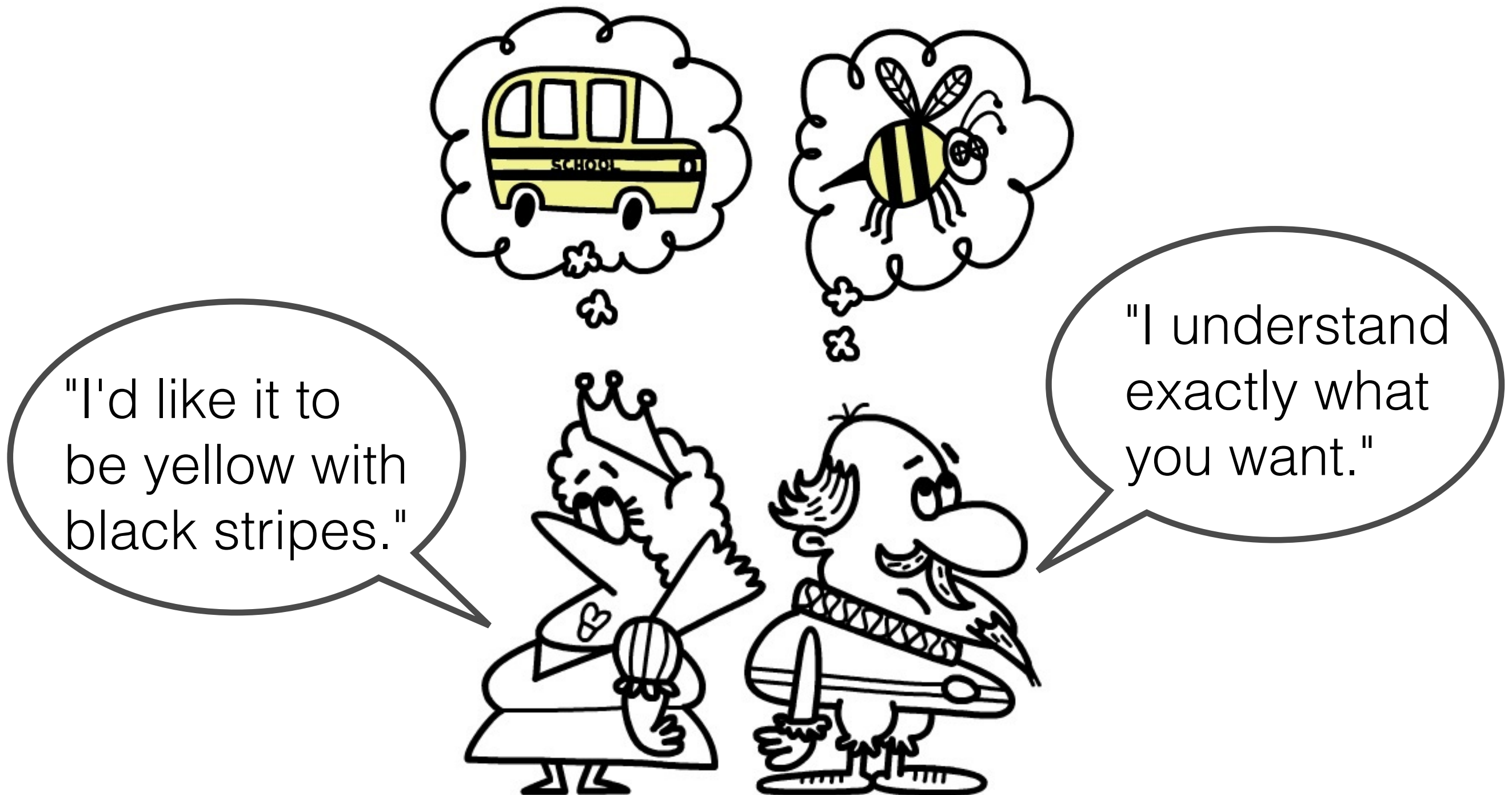


# To Document Requirements or Not To Document Requirements: That Is The Question

To document, or not to document: that is the question:  
Whether 'tis nobler in the mind to suffer missed requirements  
The slings and arrows of outrageous memory leaks and app crashes,  
Or to take arms against a sea of troubles I blame on my developers,  
And by opposing end them? To die: to sleep; To shelve the project;  
No more; and by a sleep to say we end  
The heart-ache and the thousand natural failure points of a software project  
That flesh is heir to, 'tis a consummation  
Devoutly to be wish'd. To die, to sleep; To shelve the project;  
To sleep: perchance to dream of an iPad in the hands of each of the world's children,  
For in that sleep what dreams of beautiful apps may come  
When we have shuffled off this deliverable,  
Must give us pause: there's the respect for developers and designers  
That makes calamity of so long life;  
For who would bear the whips and scorns of exploding budgets,  
The project manager's wrong, the developer's contumely,  
The pangs of despised bugs, the deliverable's delay, ...



# Documentation? Who Needs It?



# What Comprises Requirements Documentation?

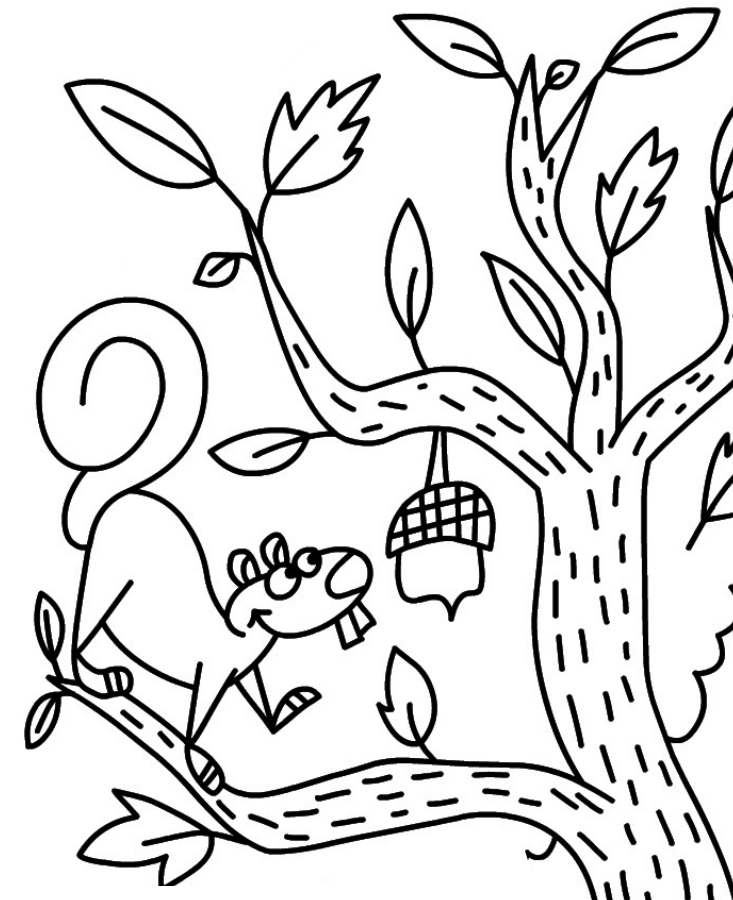
- List of Users (A.K.A. Roles)
- List of User Actions\* (A.K.A. Functional Requirements)
- Scope Document\* (A.K.A. Functional Requirements)
- Scenarios (A.K.A. User Stories, Use Cases, Functional Requirements)
- Rules (A.K.A. Business Rules, Game Rules)
- Screen Mock-Ups (A.K.A. Wire Frames, Prototype)
- Project Priorities
- Non-Functional Requirements

\* High Value



# Software Development In a Nutshell

**Developers Build Automated  
Responses To Actions Taken by  
Users**



# List of Users

- Child - Expected to be age 5-8
- Parent - Expected to play with child 100% of the time
- Teacher - Expected to manage lesson plan
- System!!! - Anticipated System Generated Events

# User Actions

## **Answer the Question: Who Does What?**

1. Child begins game play
2. Child views high score list
3. Parent selects lesson
4. Teacher changes complexity level
5. System auto-locks phone
6. System displays calendar event
7. User powers off device
8. User launches game while phone call is in progress
9. User plugs in headphones
10. System returns app to foreground

# Identifying User Actions: Know Your Platform

- Accelerometer
- Camera
- Gyro
- Speakers (Volume)
- Headphones
- AirPlay
- AirPrint
- VGA out
- GPS
- Multitouch

# Scope Document

**Scope: The list of user actions (or scenarios) to be included in a given version of your app.**

**Recommendation: prioritize your user actions (or scenarios).**

# Scenario = User Action + Response

User launches app while phone call is in progress

1. Screen gracefully adjusts to status bar height change.
2. If game play was underway during last use of app, then game play is paused and user is notified of this with an alert.
3. While call remains active, sound effects are not played

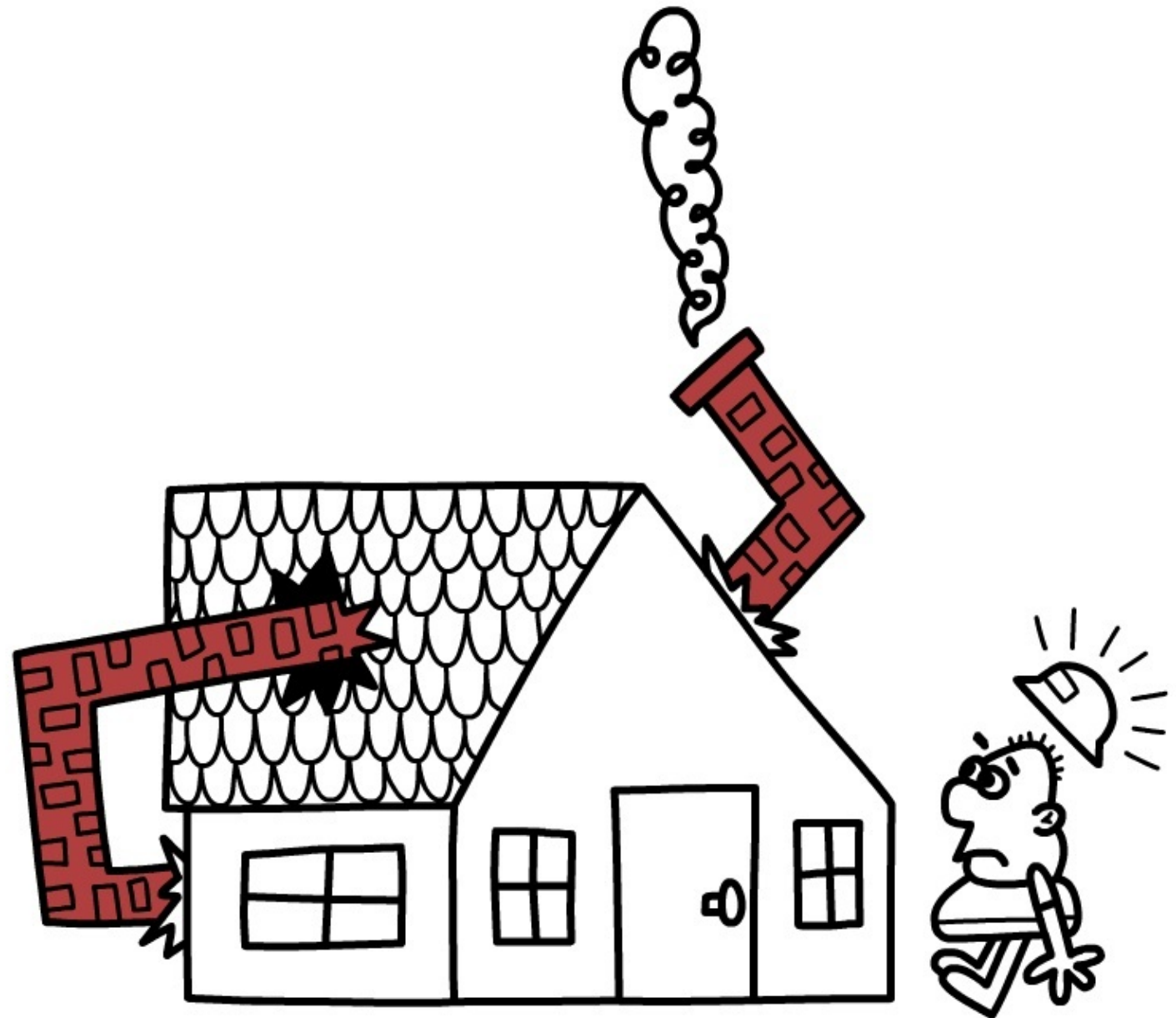
# Rules: Conditional Logic

If a given condition is met,  
then ...



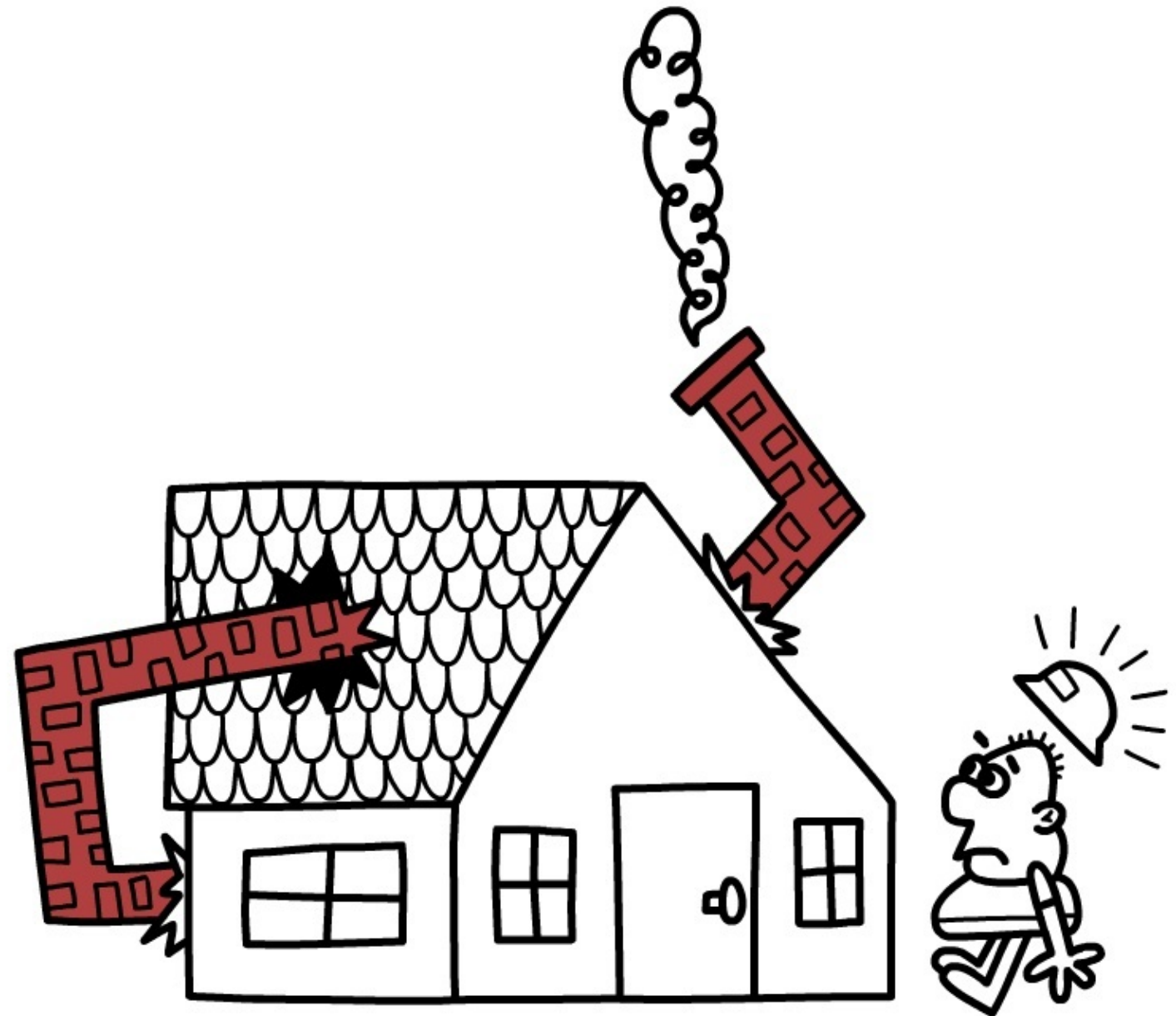
# Summary: Help Developers Plan Ahead

- By Providing Them a List of User Actions.
- By Providing Them a List of Detailed Scenarios.
- By Defining Scope for 2 Or More Versions of Your App

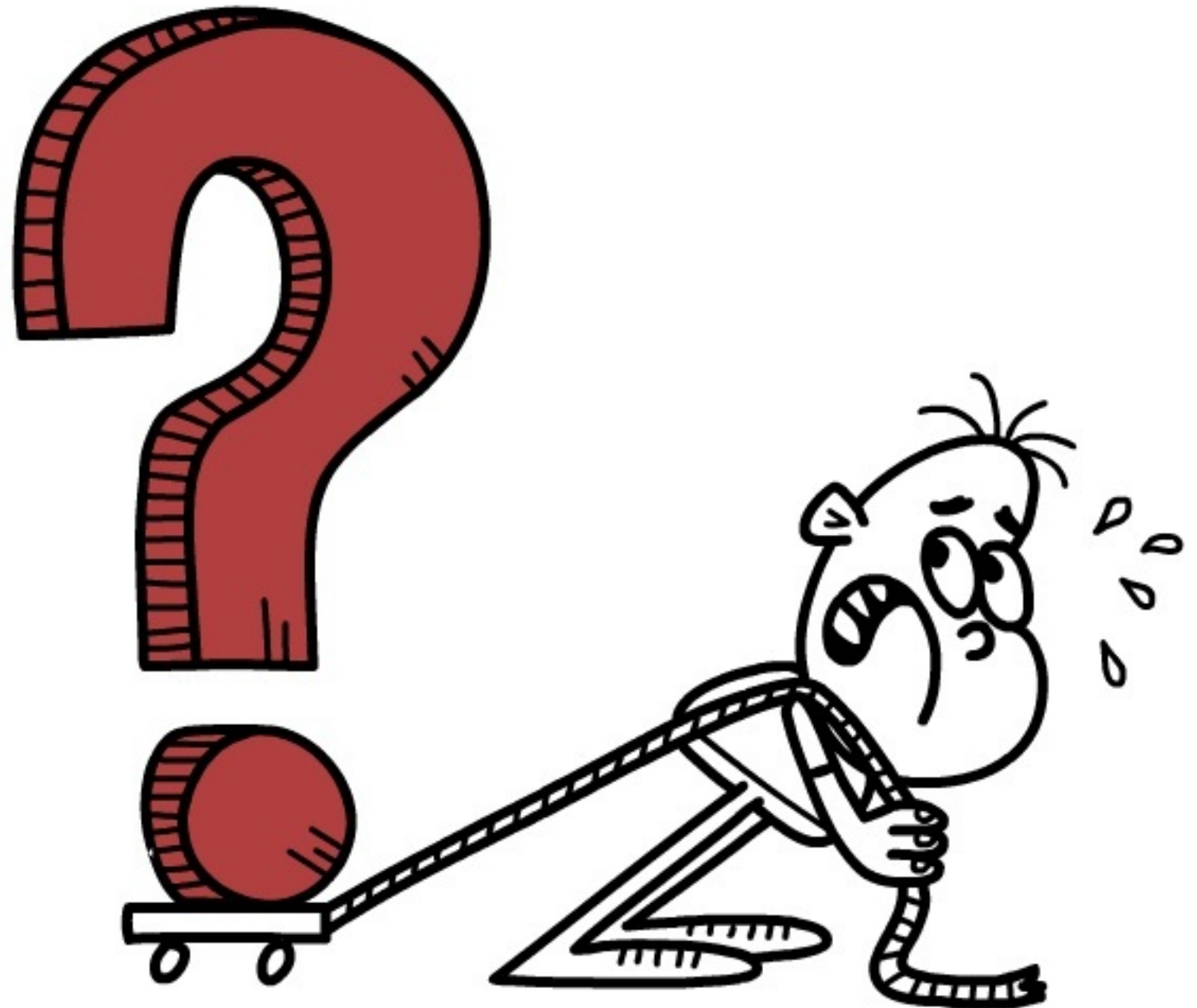


# Summary: Help Developers Plan Ahead

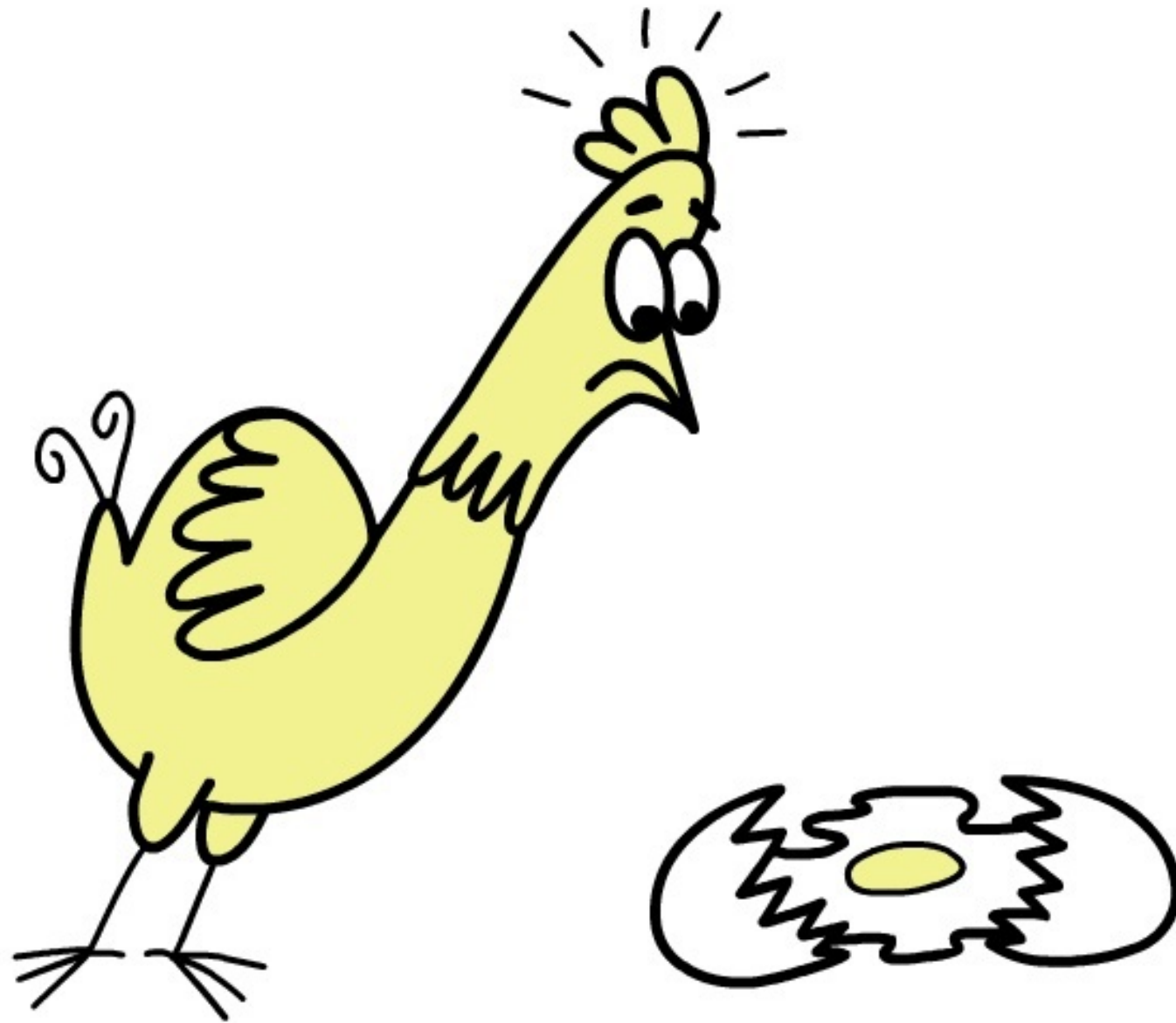
- Lower Overall Costs
- Lower Chance of Overlooking Important Requirements
- Raise Likelihood of Completing Your Project On-Time



# Questions?



# Breakout Groups



1. Functional Requirements
2. Non-Functional Requirements
3. Prototyping Screen Mock-Ups
4. Learning Obj C and Xcode
5. Outsourcing Development